****

本科毕业论文

|  |  |
| --- | --- |
| 题目: | ping程序实现 |
| Title: | Implementation of ping program |

|  |  |
| --- | --- |
| 姓 名: | 朱进 |
| 学 号: | 2220191653 |
| 学 院: | 信息科学技术学院 |
| 专业年级班级: | 网络工程2019-1 |
| 指导教师: | 刘卫江 教授 |
| 第二导师: |  |
| 完成日期: | 2023年6月 |

**毕业论文原创性声明**

本人郑重声明：所呈交的毕业论文《ping程序实现》，是本人在导师的指导下，独立进行研究工作所取得的成果。除论文中已经注明引用的内容外，对论文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本论文中不包含任何未加明确注明的其他个人或集体已经公开发表或未公开发表的成果。本声明的法律责任由本人承担。

|  |  |
| --- | --- |
|  | 学生签名： |
|  | 日 期： 年 月 日 |

**毕业论文版权使用授权书**

本人完全了解并同意遵守学校有关毕业论文知识产权的规定，在校学习期间论文工作的知识产权属于大连海事大学，学校有权保留送交毕业论文的副本，向国家有关部门或机构送交论文的复印件和电子版，允许该论文被查阅，可以将本毕业论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印、或扫描等复制手段保存和汇编本毕业论文。

涉密的毕业论文在解密后使用本声明。

|  |  |
| --- | --- |
|  | 学生签名： |
|  | 日 期： 年 月 日 |
|  |  |
|  | 指导教师签名： |
|  | 日 期： 年 月 日 |
|  |  |
|  | 第二导师签名： |
|  | 日 期： 年 月 日 |

摘 要

Ping是一个网络工具，用于测试互联网协议（IP）网络上主机的可及性。它向目标主机发送一个互联网控制信息协议（ICMP）回波请求，并等待一个ICMP回波回复。测量请求到达主机和收到回复所需的时间，并显示为往返时间（RTT）。Ping是网络工程师排除网络连接问题和测量网络性能的一个有用工具，用于测试主机之间的连通性。本文基于Python网络套机制和PySide6框架，利用PySide6和Qt Designer的GUI应用程序结合PyDracula GUI模版设计了一个图形界面化的ping命令。它提供了一个现代化的界面，可以通过左侧菜单栏访问不同的页面。其中一个页面是一个ping工具，可以通过输入URL、超时时间、数据包大小和数据包数量来测试网络连接。程序使用了pings核心类来执行ping操作，并将输出重定向到一个文件中，然后将文件内容显示在GUI中。pings核心类基于原始套接字的ping实现可以直接操作网络数据包，通过发送 ICMP 报文并接收响应，实现对主机之间连通性的测试。具体实现过程包括打开原始套接字、设置 IP 头部、发送 ICMP 报文、解析 ICMP 响应等。在发送 ICMP 报文时，需要指定目标主机的 IP 地址；在接收 ICMP 响应时，需要根据协议类型和标识码等信息进行响应的处理。通过实现基于原始套接字的 ping 命令，可以实现对网络连接状况的快速测试和监控，对于网络工程师和系统管理员具有很大的帮助。本文基于PyQt的ping命令实现是一项网络编程任务，旨在开发一个具有美观界面的ping工具。该工具使用PyQt库创建GUI，并使用原始套接字编写ping命令测试IP网络上主机的可达性。通过测量请求到达主机和接收回复所需的时间，该工具可以显示往返时间（RTT）。此工具可用于网络工程师诊断网络连接问题和测量网络性能。此外，该工具还可以通过自动化任务、监视网络性能和开发自定义网络应用程序来提高网络工程师的工作效率。

**关键词：ping；套接字编程；python；PyQt/PySide**

ABSTRACT

Ping is a network tool used to test the accessibility of hosts on Internet Protocol (IP) networks. It sends a Internet Control Message Protocol (ICMP) echo request to the target host and waits for an ICMP echo reply. Measure the time it takes for a request to arrive at the host and receive a response, and display it as round-trip time (RTT). Ping is a useful tool for network engineers to troubleshoot network connectivity issues and measure network performance, used to test connectivity between hosts. This article is based on the Python network socket mechanism and the PySide6 framework, using the GUI application programs of PySide6 and Qt Designer combined with the PyDracula GUI template to design a graphical interface based ping command. It provides a modern interface that allows access to different pages through the left menu bar.One of the pages is a ping tool that can test network connectivity by entering URL, timeout, packet size, and packet count. The program uses the pings core class to perform ping operations, redirect the output to a file, and then display the file content in the GUI. The pings core class is based on the ping implementation of the original socket, which can directly operate on network data packets. By sending ICMP packets and receiving responses, it can test the connectivity between hosts. The specific implementation process includes opening the original socket, setting the IP header, sending ICMP messages, parsing ICMP responses, and so on. When sending ICMP messages, it is necessary to specify the IP address of the target host; When receiving ICMP responses, it is necessary to process the response based on information such as protocol type and identification code. By implementing ping commands based on raw sockets, it is possible to quickly test and monitor network connection conditions, which is of great help to network engineers and system administrators. The ping command implementation based on PyQt in this article is a network programming task aimed at developing a ping tool with a beautiful interface. This tool uses the PyQt library to create a GUI, and uses the original socket to write a ping command to test the reachability of the host on the IP network. The tool can display round-trip time (RTT) by measuring the time it takes for a request to arrive at the host and receive a reply. This tool can be used by network engineers to diagnose network connectivity issues and measure network performance. In addition, this tool can improve the work efficiency of network engineers by automating tasks, monitoring network performance, and developing custom network applications.

**Keywords: ping; socket programming; python; PyQt/PySide**

目 录

[第1章 绪论 1](#_Toc12764)

[1.1 背景及意义 1](#_Toc23842)

[1.2 国内外研究现状 1](#_Toc21886)

[1.3 本文的主要研究内容 2](#_Toc20494)

[第2章 系统分析 3](#_Toc32484)

[2.1 系统目标 3](#_Toc31362)

[2.2 需求分析 3](#_Toc1079)

[2.3 性能分析 4](#_Toc29340)

[第3章 系统开发平台 5](#_Toc16719)

[3.1 Linux平台 5](#_Toc4670)

[3.2 Python语言 5](#_Toc2123)

[3.3 PyQT前端 6](#_Toc19396)

[3.3 PySide前端 6](#_Toc12262)

[第4章 关键技术分析 7](#_Toc22923)

[4.1 套接字编程 7](#_Toc30062)

[4.2 ICMP协议 8](#_Toc18047)

[4.3 校验和计算 8](#_Toc22413)

[4.5 ISO七层模型 8](#_Toc76)

[4.5.1 物理层 8](#_Toc30790)

[4.5.2 数据链路层 9](#_Toc26599)

[4.5.3 网络层 9](#_Toc18377)

[4.5.4 传输层 9](#_Toc15238)

[4.5.5 会话层 10](#_Toc1812)

[4.5.6 表示层 11](#_Toc19784)

[4.5.7 应用层 11](#_Toc26751)

[4.6 数据包结构 12](#_Toc21524)

[4.7 计时和统计 12](#_Toc1431)

[4.8 本章小结 12](#_Toc20601)

[第5章 环境信息 13](#_Toc23807)

[5.1 硬件资源 13](#_Toc3484)

[5.2软件资源 13](#_Toc24622)

[第6章 项目概览 14](#_Toc18029)

[6.1 系统总体功能结构 14](#_Toc6208)

[6.2 项目结构 14](#_Toc31169)

[第7章 系统设计 15](#_Toc25148)

[7.1 主要程序 15](#_Toc3077)

[7.1.1 Ping类 15](#_Toc1645)

[7.1.2 PingResponse类 16](#_Toc2970)

[7.2 图形界面设计 17](#_Toc19408)

[7.2.1 启动程序 18](#_Toc21800)

[7.2.2 子启动程序 19](#_Toc28577)

[7.2.3 UI设计 20](#_Toc32611)

[第8章 运行效果 22](#_Toc27270)

[8.1 主程序 22](#_Toc29206)

[8.2 Ping程序页面 22](#_Toc30704)

[8.3 Ping测试1 23](#_Toc27251)

[8.4 Ping测试2 23](#_Toc14760)

[8.4 Ping测试3 24](#_Toc32505)

[结论与展望 25](#_Toc24742)

[参考文献 26](#_Toc1902)

[致谢 27](#_Toc19865)

# 第1章 绪论

基于PyQt的ping命令实现是一项网络编程任务，旨在开发一个具有美观界面的ping工具。该工具使用PyQt库创建GUI，并使用ping命令测试IP网络上主机的可达性。通过测量请求到达主机和接收回复所需的时间，该工具可以显示往返时间（RTT）。此工具可用于网络工程师诊断网络连接问题和测量网络性能。此外，该工具还可以通过自动化任务、监视网络性能和开发自定义网络应用程序来提高网络工程师的工作效率。

在本绪论中，我们将介绍ping命令的背景和作用，以及PyQt库的基本概念和用途。我们还将讨论ping命令的实现方式和GUI设计，以及如何使用PyQt库创建GUI。最后，我们将探讨ping命令实现的优点和局限性，并提出未来的研究方向。

## 1.1 背景及意义

基于PyQt的ping命令实现是一项网络编程任务，旨在开发一个具有美观界面的ping工具。Ping是一种网络实用工具，用于测试Internet Protocol（IP）网络上主机的可达性。它向目标主机发送Internet Control Message Protocol（ICMP）回显请求，并等待ICMP回显回复。测量请求到达主机和接收回复所需的时间，并将其显示为往返时间Round-Trip Time（RTT）。Ping是网络工程师诊断网络连接问题和测量网络性能的有用工具。

PyQt是一个用于创建GUI应用程序的Python库。它提供了一组丰富的GUI组件和工具，使开发人员能够轻松地创建具有吸引力和交互性的应用程序。使用PyQt库创建GUI，可以使ping工具更加易于使用和美观[1]。

基于PyQt的ping命令实现的意义在于，它可以提高网络工程师的工作效率。通过自动化任务、监视网络性能和开发自定义网络应用程序，网络工程师可以更快地诊断和解决网络问题。此外，基于PyQt的ping命令实现还可以为网络工程师提供一个可视化的工具，使他们更容易地理解和分析网络数据。未来的研究方向可能包括将ping工具与其他网络工具集成，以提供更全面的网络分析功能。

## 1.2 国内外研究现状

在国内外，已经有很多关于ping命令实现的研究。例如，国外的PingPlotter和国内的PingER都是基于ping命令的网络性能分析工具。这些工具使用ping命令测试网络连接，并将结果可视化为图表或图形。此外，还有一些研究关注ping命令的优化和改进，以提高其性能和可靠性。

在PyQt方面，该库已经被广泛应用于GUI应用程序的开发。许多开发人员使用PyQt创建各种类型的应用程序，包括网络工具。例如，Wireshark是一种流行的网络协议分析工具，它使用PyQt创建GUI。此外，还有一些研究关注PyQt库的优化和改进，以提高其性能和可靠性。

基于原始套接字的PyQt的ping命令实现的意义在于，它可以提高网络工程师的工作效率。通过自动化任务、监视网络性能和开发自定义网络应用程序，网络工程师可以更快地诊断和解决网络问题。此外，基于PyQt的ping命令实现还可以为网络工程师提供一个可视化的工具，使他们更容易地理解和分析网络数据。未来的研究方向可能包括将ping工具与其他网络工具集成，以提供更全面的网络分析功能，并进一步优化和改进PyQt库的性能和可靠性[2]。

在实现方面，使用原始套接字可以更好地控制ping命令的实现过程，从而提高其性能和可靠性。同时，使用PyQt库可以使ping工具更加易于使用和美观。具体而言，我们将介绍ping命令的背景和作用，以及PyQt库的基本概念和用途。我们还将讨论ping命令的实现方式和GUI设计，以及如何使用PyQt库创建GUI。最后，我们将探讨ping命令实现的优点和局限性，并提出未来的研究方向。本文旨在为网络工程师提供一个功能强大、易于使用和美观的网络工具，有助于他们更快地诊断和解决网络问题。同时，本文也为其他开发人员提供了一个使用PyQt库创建GUI应用程序的示例。

综上所述，基于PyQt的ping命令实现是一个有前途的研究方向。通过结合ping命令和PyQt库，可以开发出一个功能强大、易于使用和美观的网络工具，有助于网络工程师更快地诊断和解决网络问题。未来的研究方向可能包括将ping工具与其他网络工具集成，以提供更全面的网络分析功能，并进一步优化和改进PyQt库的性能和可靠性。

## 1.3 本文的主要研究内容

本文的主要研究内容是基于PyQt的ping命令实现。具体而言，我们将介绍ping命令的背景和作用，以及PyQt库的基本概念和用途。我们还将讨论ping命令的实现方式和GUI设计，以及如何使用PyQt库创建GUI。最后，我们将探讨ping命令实现的优点和局限性，并提出未来的研究方向。本文旨在为网络工程师提供一个功能强大、易于使用和美观的网络工具，有助于他们更快地诊断和解决网络问题。同时，本文也为其他开发人员提供了一个使用PyQt库创建GUI应用程序的示例[3]。

## 1.4 本文的组织结构

根据论文的内容要求，论文的各章节安排如下：

第一章：绪论。主要介绍本文的研究背景及长流识别的意义、国内外的研究

现状、论文的主要工作和组织结构。

第二章：系统分析。主要介绍该系统设计目标，设计操作对象和操作流程，最后进行性能评估。

第三章：系统开发平台。介绍了本项目所使用到的各种软件技术与平台。

第四章：关键技术分析。分析了网络套接字编程对于项目的重要性，介绍OSI网络七层模型，从数据链路层到应用层具体分析其作用，理解和使用ICMP协议构造Ping数据包。

第五章：环境信息。介绍本项目运行的硬件基础设施环境。

第六章：项目概览。介绍了本项目的整体架构。解释每个重要目录和文件和作用。让读者对本项目有一个整体结构的认识。

第七章：系统设计。介绍了本项目的核心程序和图形界面。介绍了发送和接受ping数据包的核心类以及其成员函数。解释了其工作原理和整体的结构。

第八章：运行效果。介绍了程序运行时的操作和运行时的结果，并给出解释和截图。

# 第2章 系统分析

## 2.1 系统目标

设计和实现基于原始套接字的 ping 命令的系统目标可能包括以下内容：

向目标主机发送 ICMP 回显请求消息并测量响应消息到达所需的时间，以确定目标主机的网络延迟或可达性。

允许将目标主机的 IP 地址或域名指定为 ping 命令的命令行参数。

使用原始套接字发送 ICMP 回显请求和接收 ICMP 回显响应消息，这样可以更好地控制消息的内容和传入消息的过滤。

包括其他功能，例如设置 IP 标头的 TTL 字段，提供有关发送/接收/成功/丢失的数据包数量的统计信息，并允许自定义 ICMP 消息有效负载的内容。

基于原始套接字的 ping 命令的目标是提供一种比操作系统提供的标准 ping 命令更灵活和可定制的方式来执行网络测量和故障排除[4]。

## 2.2 需求分析

要设计和实现基于原始套接字的ping命令，我们首先需要了解系统的要求和约束。

ping 命令用于测试网络中两台主机之间的连通性。 它向目标主机发送一个 ICMP 回声请求数据包，并等待一个 ICMP 回声应答数据包。 计算数据包的往返时间 (RTT) 并将其显示给用户。

要实现基于原始套接字的ping 命令，我们需要使用Python 套接字模块创建原始套接字并向目标主机发送ICMP 回显请求数据包。 我们还需要处理 ICMP echo reply 数据包并计算每个数据包的 RTT[5]。

ping命令要求如下：

1. 该命令应将主机名或 IP 地址作为输入。

2. 该命令应向目标主机发送ICMP 回显请求数据包。

3. 该命令应等待来自目标主机的 ICMP 回显回复数据包。

4. 该命令应计算每个数据包的 RTT。

5. 命令应向用户显示结果。

基于这些需求，我们可以使用Python socket模块设计并实现一个ping命令。 实施应包括以下步骤：

1. 使用带有 AF\_INET 地址族、SOCK\_RAW 套接字类型和 IPPROTO\_ICMP 协议的 socket.socket() 函数创建原始套接字。

2. 使用 struct.pack() 函数打包 ICMP 回显请求数据包。

3、使用socket.sendto()函数向目标主机发送ICMP回显请求包。

4. 使用 socket.recvfrom() 函数等待 ICMP 回显回复数据包。

5. 计算数据包的RTT。

6. 对指定数量的数据包重复步骤 2-5。

7. 向用户显示结果。

### 2.2.1 ping核心程序时序图

ping命令核心实现流如图2-1所示。

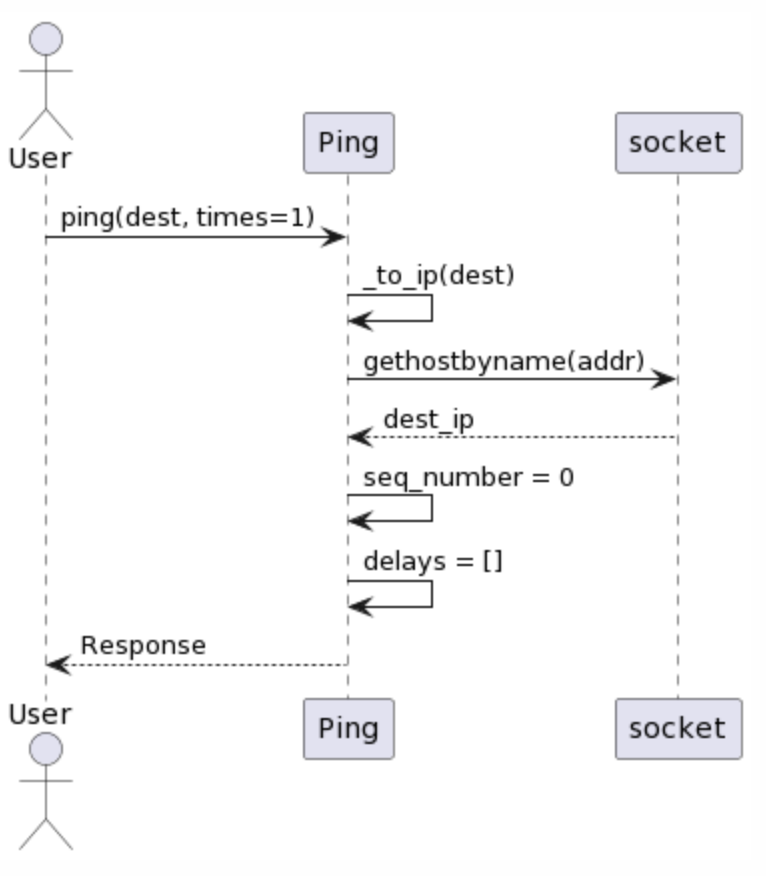


图 2-1 核心实现过程

这定义了用户参与者使用 dest 参数和默认时间参数 1 向 Ping 对象发送 ping 消息。Ping 对象然后调用其 \_to\_ip 方法来解析目标地址，然后调用 socket.gethostbyname 以获取IP地址。然后将 dest\_ip 值存储在 Ping 对象中，并初始化 seq\_number 和 delays 变量。

ping响应过程如图2-2所示。

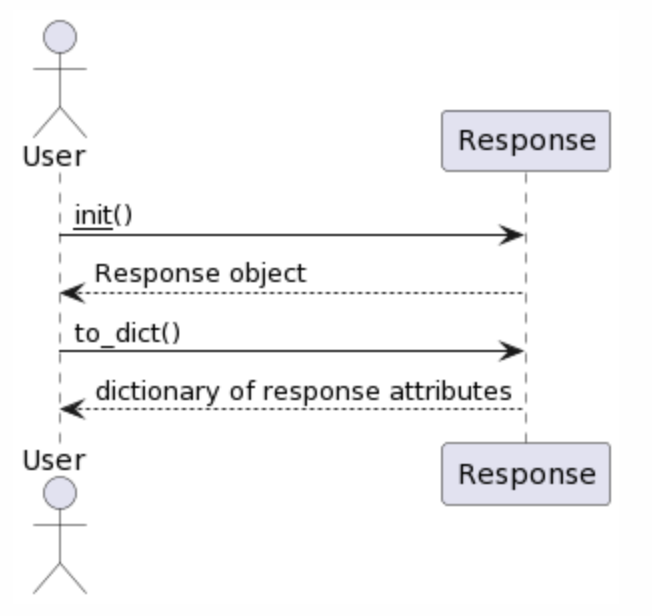


图 2-2 ping响应

这定义了用户参与者通过调用其 \_\_init\_\_ 方法来创建 Response 对象，然后调用 to\_dict 方法来获取响应属性的字典。

### 2.2.2 图形界面时序图

图形界面逻辑如图2-3所示。

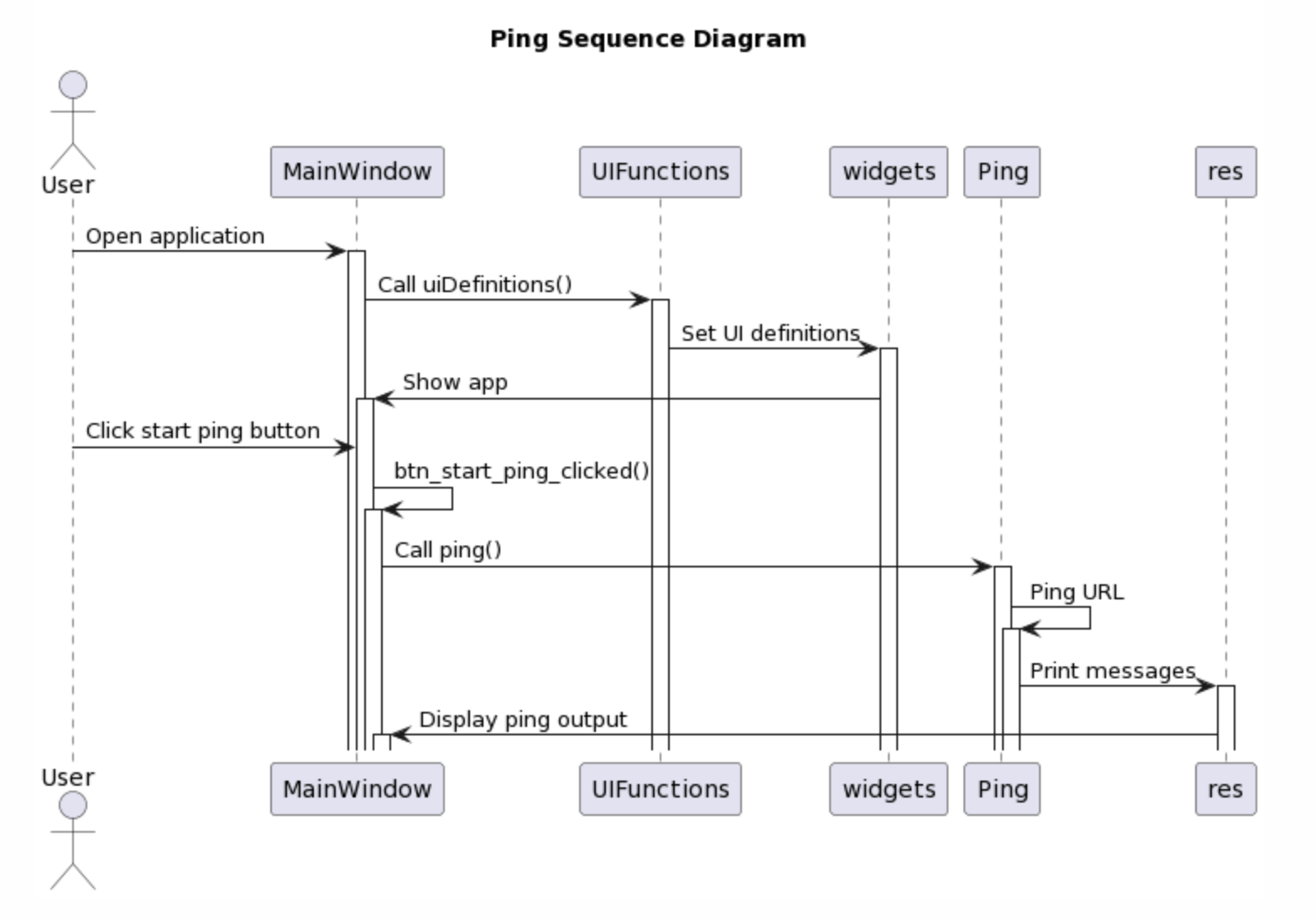


图 2-3 图形界面逻辑

此图显示了当用户打开应用程序并单击启动 ping 按钮时用户、主窗口、UI 功能、小部件和 Ping 类之间的交互顺序。它还显示了每个对象在交互过程中的激活情况。

首先，用户通过与 MainWindow 对象交互打开应用程序。然后 MainWindow 对象调用 UIFunctions 模块中的 uiDefinitions() 函数来设置小部件的 UI 定义。然后激活小部件对象并向用户显示应用程序。

当用户单击开始 ping 按钮时，将从 MainWindow 对象调用 btn\_start\_ping\_clicked() 函数。然后此函数调用 Ping 类中的 ping() 函数来启动 ping 过程。 Ping 类然后 ping 指定的 URL 并将消息打印到 rest 对象。最后，MainWindow 对象向用户显示 ping 输出。

## 2.3 性能分析

因为 Python 的性能可能取决于各种因素，例如正在执行的操作类型、数据集的大小、代码的实现 ，以及用于执行代码的硬件。

然而，关于 Python 性能的一些一般性观察包括：

Python 是一种解释型语言，因此对于某些操作，它可能比 C 或 C++ 等编译型语言慢。

Python 的动态类型也会影响性能，因为类型检查发生在运行时。

由于性能优化，Python 3 往往比 Python 2 更快，例如默认进行某些操作，例如整数除法底除法。

在 Python 中使用原始套接字的性能可能受到各种因素的影响，例如发送和接收数据包的大小和频率、运行代码的系统的处理速度和硬件以及代码本身的实现。

优化 Python 和原始套接字性能的一些技巧包括：

使用适当的缓冲区大小从套接字接收数据，以最大限度地减少内存使用并减少数据包丢失的机会。 默认缓冲区大小可能不适合较大的传入数据包。

通过避免对数据包负载进行不必要的操作，或将繁重的处理任务委托给单独的线程或进程，最大限度地减少数据包数据的处理时间[6]。

对丢失或丢失的数据包实施错误处理和重试，以避免过度延迟和重试，并提高通信的整体性能。

此外，使用像Scapy这样的库可以简化构建和发送数据包的过程，还可以提供数据包拦截和修改等附加功能，可以提高通信的效率和灵活性。

总体而言，Python 是一种用途广泛且使用广泛的语言，具有许多可用于优化其性能的库和工具。还有许多优化 Python 代码的技术，例如使用列表理解、避免不必要的循环以及利用排序算法和数据结构等内置功能[7]。

# 

# 第3章 系统开发平台

## 3.1 Linux平台

Linux 是一种基于类 Unix 内核的免费开源操作系统。 它于 1991 年由 Linus Torvalds 首次发布，此后成为世界上最受欢迎的操作系统之一，为从服务器到智能手机的一切提供支持。

Linux 的主要优势之一是其灵活性和定制选项。 它是高度可配置的，可以进行定制以满足从简单的桌面环境到复杂的企业级系统的广泛需求。 Linux 还有大量可用的软件选项，可以使用 apt 或 yum 等包管理器轻松安装。

此外，Linux 以其稳定性和安全性功能而闻名。 因为它是开源的，代码会受到开发人员和用户的不断审查，这有助于快速识别和修复漏洞。 Linux 还使用用户级权限和其他安全措施来防止恶意软件和未经授权的访问。

Linux 的另一个特点是它的命令行界面，它允许用户使用基于文本的命令与系统交互。 对于高级用户和系统管理员来说，这可能是一个强大的工具，并且经常用于脚本和自动化任务。

Linux 还可以根据其图形用户界面 (GUI) 进行自定义。 Linux 有许多不同的桌面环境，每个都有自己的外观、感觉和功能。 流行的选择包括 GNOME、KDE 和 XFCE[8]。

## 3.2 Python语言

Python是一种高级编程语言，由Guido van Rossum于1989年创建。Python的设计目标是让代码简单易读、易于维护和扩展，同时具有丰富的标准库和第三方库。

Python在许多领域，比如Web开发、数据科学、机器学习和人工智能方面都有广泛的应用。Python的语法简洁明了，且容易学习，具有良好的可读性和可维护性。Python支持多种编程范式，包括面向对象、函数式编程和过程式编程。

Python具有许多强大的标准库，在网络编程、多线程编程、图形用户界面编程、正则表达式和加密等方面提供了丰富的支持。此外，Python拥有大量的第三方库和框架，可以扩展Python的功能和应用范围[9]。

Python语言具有跨平台的特性，可以在多种操作系统下运行，包括Windows、Linux和macOS等。Python还支持多种解释器，包括CPython、Jython和IronPython等。

总之，Python是一种优秀的高级编程语言，具有简洁易读、易于学习、跨平台和丰富的库支持等特点，这使得它在许多领域中成为开发者的首选。

## 3.3 PyQT前端

PyQt 是流行的 Qt 应用程序框架的一组 Python 绑定。 Qt 是一个用 C++ 编写的 GUI 工具包，用于开发具有单一代码库的跨平台应用程序。 PyQt 允许开发人员使用 Qt 框架在 Python 中编写 GUI 应用程序。

PyQt 提供了范围广泛的类和工具，使图形用户界面的编程更加容易和高效。 它具有一系列可自定义的小部件，包括按钮、表单、菜单和对话框，以及对动画、图形和多媒体、数据库和线程等高级功能的支持。

PyQt 应用程序可以运行在各种操作系统，如 Windows、Linux 和 macOS，以及移动操作系统，如 Android 和 iOS，使用 PyQt 的移动版本，PyQt for mobile。 PyQt 还支持多种编程范式，包括过程式、函数式和面向对象编程。

PyQt 有一个活跃和支持性的社区，提供大量的文档、教程和论坛来帮助初学者和有经验的程序员。 PyQt 在两种许可下可用 - GPL 和商业许可。 GPL 许可证使 PyQt 成为开源开发的绝佳选择，而商业许可证允许开发人员在专有项目中使用 PyQt[10]。

## 3.3 PySide前端

PySide6是一个跨平台GUI工具包Qt6的Python绑定。它为Python开发者提供了对完整的Qt6 API的访问，包括部件、图形、多媒体、网络等，使他们能够创建强大而复杂的桌面和移动应用程序。PySide6兼容广泛的平台，包括Windows、macOS、Linux和Android。

PySide6的一些主要特点包括：

1. 支持最新的Qt6 API和功能。
2. 在所有支持的平台上具有原生的外观和感觉。
3. 与PyQt6完全兼容，允许开发者在两个绑定之间切换而无需修改代码。
4. 全面的文档和资源，包括一个用于可视化UI设计的Qt Designer工具。
5. 支持Qt Widgets和Qt Quick，使开发者可以灵活地选择适合他们项目的最佳方法。

PySide6为Python开发者提供了一个强大而灵活的平台，让他们可以轻松地创建跨平台的GUI应用程序。

# 第4章 关键技术分析

## 4.1 套接字编程

python套接字编程

Python Socket编程是一种基于Socket通信协议的编程技术，它使得程序可以通过网络传输数据。Socket通信协议是网络编程领域中最重要的一个协议之一，可以利用Socket协议在两个计算机之间进行数据传输，实现进程之间的通信。在Python中，通过使用标准库中的socket模块，可以轻松地进行Socket编程。具体来说，通过socket模块可以实现TCP和UDP套接字编程，包括创建套接字、绑定套接字、监听连接请求、接受连接请求、发送和接收数据等操作，同时还包括异常处理等内容。

在Socket编程中，TCP和UDP套接字不同，它们分别对应于不同的网络通信协议。TCP套接字提供了一种可靠的、面向连接的数据传输服务，它与传输层的TCP协议配合工作，可以保证数据的完整、可靠性和有序性。而UDP套接字则提供了一种不可靠的、无连接的数据传输服务，它与传输层的UDP协议配合工作，可以快速地传输数据，但不保证数据的完整性、可靠性和有序性。

通过Python Socket编程，可以实现各种网络应用程序，例如聊天工具、文件传输工具、远程控制工具、Web服务器、SMTP邮件发送工具等等。Socket编程是一项非常重要的技术，掌握它可以帮助开发者快速实现各种网络通信应用。

套接字编程是一种在计算机网络中进行通信的方法，它允许进程在不同的计算机之间交换数据。套接字是一个使用 TCP/IP 协议的通信端点，可以用于建立网络连接并传输数据。在 Python 中，使用 socket 模块可以进行套接字编程。

在套接字编程中，通常有两种主要类型的套接字：客户端套接字和服务器套接字。客户端套接字用于与远程服务器进行通信，而服务器套接字用于接收客户端连接请求并提供服务。

Python 的 socket 模块提供了创建和操作套接字所需的函数。通过创建套接字对象、指定主机地址和端口号以及使用函数进行连接和数据传输，可以实现套接字编程。除此之外，socket 模块还提供了其他函数和常量，比如选择事件、多线程编程和错误处理。

在进行套接字编程时，需要注意网络连接和通信的稳定性和安全性。可以使用加密协议、认证机制和数据加密等技术来提升网络安全。此外，还可以处理网络错误和异常，防止程序崩溃。

套接字编程是一种重要的网络通信方式，可以用于实现各种网络应用程序，比如网络聊天、文件传输、远程控制和远程协作等等。Python 的 socket 模块提供了强大且易于使用的工具，使得套接字编程变得更加简单和高效。

## 4.2 ICMP协议

ICMP 数据包具有特定的结构，包括标头和数据负载。 标头包含有关数据包类型（回显请求或回显回复）、数据包 ID 和校验和的信息，以确保数据完整性。 数据负载可以是任意数据，但通常是一条短消息。

校验和用于通过验证数据在传输过程中未被破坏来确保数据完整性。 校验和是使用特定算法在整个数据包（标头和数据有效负载）上计算的。

网络设备使用 ICMP 协议来传达有关网络状况的错误消息和操作信息。 ICMP 消息通常由路由器或其他网络设备生成，以响应需要注意的错误或其他情况。 例如，如果路由器收到它无法转发的数据包，它会生成一条 ICMP 消息，通知发送方该数据包未送达。

要衡量 ping 命令的性能，您需要测量每个回显请求的往返时间 (RTT) 并计算统计信息，例如最小、最大和平均 RTT，以及丢包率。可以使用Python的timeit模块来测量ping函数的执行时间。

## 4.3 校验和计算

校验和是在数据传输过程中用于检测数据错误的一种方法。在计算校验和时，通常采用累加和或者异或和的方法。具体来说，累加和是将数据逐个字节相加，再将结果通过特定的算法进行处理得到最终的校验和；异或和是将数据逐个字节进行异或操作，再将结果进行处理得到最终的校验和。

在计算校验和时，通常需要将校验和添加到原有的数据中一起传输，接收方在接收到数据之后重新计算校验和，然后与传输时发送的校验和进行比较，判断数据是否发生了错误。如果接收方计算出的校验和与发送方传输的校验和不一致，则说明数据发生错误，需要重新传输或者采取其他的纠错措施。

在计算校验和时，为了提高校验的准确性和可靠性，通常会采用多次计算和校验的机制，例如在TCP协议中就采用了多项式校验和（CRC）的算法，通过多轮的异或计算和余数计算，可以实现在网络传输中高效地检测数据是否发生错误。 校验和是从数据块（例如文件或消息）计算得出的值，以检测在传输或存储过程中可能引入的错误。 校验和通常是使用数学算法计算的固定大小的字节串。

一种用于计算校验和的常用算法是 Internet 校验和，它用于 TCP/IP 协议族。 Internet Checksum 是一种简单的算法，涉及迭代数据字节并将它们相加，溢出被环绕。 然后对结果进行补码并以 16 位整数的形式返回

## 4.5 ISO七层模型

### 4.5.1 物理层

IOS七层模型的物理层是该模型的第一层，负责实现数据的物理传输。物理层处理数字信号和物理媒体，将数据转换为二进制比特流，并将其发送到网络。物理层定义了各种特性，例如传输媒介、接口规范、电气特性和数据速率。在网络通信中，物理层要考虑如何准确传输数据和如何实现传输性能的平衡。

### 4.5.2 数据链路层

数据链路层是计算机网络七层模型的第二层，负责将物理层传递的比特流组装成帧，进行数据的传输和接收。它在物理连接上建立逻辑连接，并提供了一种检测和纠正数据传输错误的机制。

数据链路层的主要功能包括：

1. 将网络层传来的数据包进行分割，并在每一部分前添加特定的头尾标识符，形成数据帧。
2. 进行差错检测和纠正，确保数据的可靠传输。
3. 进行流量控制，避免因发送速度过快导致接收方无法处理的情况。
4. 进行地址识别，管理物理地址和 MAC 地址，以便正确传输数据帧。
5. 数据链路层建立了两个相邻节点之间的数据链路（即逻辑连接），使数据经过数据链路层的处理后可以可靠地从发送方传输到接收方。在数据链路层中，广泛使用的协议包括 Ethernet 和 Wi-Fi 等。

总之，数据链路层是计算机网络中非常重要的一层，实现了将物理连接上的数字信号转换为帧的功能，并提供了许多实用的机制保证数据传输的可靠性和流畅性。

### 4.5.3 网络层

ISO网络层是ISO/OSI模型的第三层，也称为网络层。网络层的作用是为不同的计算机网络之间提供通信服务，以实现数据包在不同的网络之间的传输。

网络层的主要功能包括：

寻址功能：通过网际协议地址（IP地址）为数据包选择最佳的传输路线。

路由选择：网络层可以使用路由算法找到最短的路径或最具效率的路线，以便数据包能够正确、快速地到达目标主机。

分组封装：网络层将需要传输的数据分割为小数据包，并在每个数据包的头部加上网络层的协议头，以实现不同的网络层协议之间互相通信。

拥塞控制：网络层负责监控并控制数据包的传输速率，以避免网络拥塞，以及避免由于传输率过慢导致的资源浪费问题。

### 4.5.4 传输层

ISO (International Organization for Standardization) 的传输层是 OSI 参考模型的一部分。传输层是 OSI 参考模型的第四层，负责在不同主机之间提供可靠的数据传输。该层的协议可以基于面向连接的和无连接的传输方式，常用协议有传输控制协议 (TCP) 和用户数据报协议 (UDP)。

ISO 传输层的主要功能包括：

1. 分段和重组数据：将来自应用层的数据进行分段以适应网络传输，同时在接收方重组数据，使其可以进入应用层。
2. 差错恢复：通过序列号和确认消息，使数据能够可靠传输，即在数据传输过程中能够自动进行重传等机制，保证数据的完整性和正确性。
3. 数据流量控制：根据不同的情况，控制数据的传输速率，避免网络拥塞和资源浪费。
4. 数据传输信道管理：通过端口号与应用层服务的关联，以确保分段、传输、重组顺序执行。

在 ISO 参考模型中，协议栈的传输层使用源端到目标端端到端连接，并提供了一些对应用层服务和网络层协议相关的传输质量（如时延、带宽）的保证。

### 4.5.5 会话层

会话层是计算机网络中的第五层，也是OSI参考模型中的一部分。会话层提供了建立、管理和结束会话的机制，以便不同的应用程序能够在网络上通信。

会话层的主要功能包括：

1. 建立和管理会话：会话层通过建立和管理会话来处理可靠通信的需求。这包括与应用程序配合确定会话内的数据交换规则、建立和认证用户许可和权限等。
2. 数据交互控制：会话层能够使应用程序并行处理多个对话，控制会话顺序，确保数据的正确接收，以及处理并发数据请求和响应，实现复杂的数据通信逻辑。
3. 会话恢复：在与网络通讯过程中，对于未完成的会话，会话层通过记录信息，以便在系统崩溃或发生错误时进行恢复，保证通讯的顺畅进行。
4. 安全控制：在会话层，可以使用加密、认证等机制来保证通信的机密性和安全性。 session层还可以针对网络延迟和故障等情况进行调整会话的优先级。
5. 常见的会话层协议有如下几种：网络终端协议（NCP）、AppleTalk的数据流管理协议（DMP）和传输控制协议（TCP）等。

总之，会话层通过建立、管理和结束会话，为不同的应用程序在网络上通信提供了机制和支持。会话层的功能包括数据交互控制、会话恢复和安全控制等，为高可靠性和安全的通讯提供了建立和保障

### 4.5.6 表示层

表示层是计算机网络中的第六层，也是OSI参考模型的一部分。表示层的主要任务是定义数据的格式和编码方式，确保一个系统中的应用程序能够正确地理解另一个系统中的数据格式。

表示层的功能主要包括以下几点：

1. 数据格式化：表示层将数据从应用层传送到会话层之前进行格式化。数据格式化可以包括数据压缩、数据加密、数据转换（如将各种编码方式相互转化）、数据压缩、图形展示等操作。
2. 数据加密：表示层可以使用加密算法或其他机制来保护数据的机密性。
3. 数据描述：表示层使用语法和语义规则来描述数据格式，确保不同系统中的应用程序可以互相理解对方的数据格式。
4. 常见的表示层协议有如下几种：数据表示协议（SDP）、外部数据表示协议（ODP）、字节流协议（BTP）、数据交换格式（DXF）等。

总之，表示层是 OSI 参考模型中的重要层级，负责数据格式及编码方式的定义。表示层的主要功能包括数据格式化、加密和描述，以确保别的系统中的应用程序能够接收并理解数据的格式。

### 4.5.7 应用层

应用层是计算机网络中的最上层，也是OSI参考模型的一部分。应用层协议是应用程序和网络之间的接口，能够提供特定的网络服务，例如文件传输、电子邮件、远程访问、Web访问和流媒体服务等。

应用层的主要功能包括：

1. 应用协议：应用层定义了许多应用程序可以使用的标准协议，如HTTP、SMTP、POP3、FTP等，这些协议规定了应用程序之间的通信方式和数据格式。
2. 数据格式化：应用层将数据转换为特定的数据格式，以便于在网络上传输和接收。例如，将文本数据转换为HTML格式。
3. 数据打包：应用层会将对应用程序所需的其他信息打包到数据中，以便进行数据传输。
4. 用户接口：应用层的应用程序与用户之间的界面，提供了简单直接的用户友好入口。

常见的应用层协议有如下几种：超文本传输协议（HTTP）、文件传输协议（FTP）、电子邮件协议（SMTP、POP3、IMAP）、DNS、Telnet以及许多P2P协议等。

总之，应用层是 OSI 参考模型的最上层，负责定义了许多应用程序能够使用的标准协议和接口，实现了对各种应用程序的支持，为各种网络应用提供了基础服务。

## 4.6 数据包结构

数据包是计算机网络中传输的基本数据单位，通常由多个字节构成。数据包结构包括以下几个方面：

标识符：数据包中包含一个唯一的标识符，用于标识数据包的类型和目的。例如，TCP 和 UDP 数据包中包含端口号用于唯一标识该数据包所用的协议。

数据载荷：数据包中包含要传输的信息。载荷的类型和格式取决于所使用的协议。

报头：数据包中的报头包含诸如源和目标地址、数据长度等元数据。这些元数据是用来控制和管理数据包传输的，不包括数据本身。

校验和：许多协议在数据包中维护了一个校验和，用于验证数据包是否完整且正确。

尾部信息：一些协议在数据包的尾部添加了一些元数据，例如 TCP 协议在数据包尾部添加了一个用于终止连接的标识符。

数据包是计算机网络中传输数据的基本单位，数据包结构包括标识符、数据载荷、报头、校验和和尾部信息等。每个协议的数据包结构不同，具体取决于协议的实现。

## 4.7 计时和统计

IP协议可以进行数据包计时和统计。其中计时主要用于判断数据包在传输过程中是否丢失或超时。当一个数据包被发送出去时，它会在包头中记录发送时间戳，而当数据包到达接收方时，接收方会将当前时间戳保存在包头中。发送方收到接收方返回的确认数据包时，会计算包的往返时间（RTT）来确定网络的延迟时间和数据包到达的时间。

IP协议还可以进行各种类型的统计工作，以帮助管理者更好地了解网络状况。例如，可以统计网络中收到的数据包数量、发送的数据包数量、接收丢失的数据包数量等。

IP协议支持数据包计时和统计功能，这些功能对于网络管理和故障排查非常有帮助。

## 4.8 本章小结

本章主要介绍了公式、数字、英文、符号、物理量及单位、上标和下标等特殊格式的编辑方法以及所用到的关键技术和基本概念。

# 第5章 环境信息

## 5.1 硬件资源

本系统的硬件资源为计算机，具体配置如表5.1.1所示

表5.1.1硬件配置

|  |  |
| --- | --- |
| CPU | Intel CORE i7 |
| 内存 | 8GB |
| 显卡 | 1050ti |

## 5.2软件资源

本系统的硬件资源为计算机，具体配置如表5.2.1所示

表5.2.1软件环境

|  |  |  |
| --- | --- | --- |
|  | 名称 | 版本 |
| 运行操作系统 | Ubuntu | 22.04 |
| 代码编辑器 | Vim/vscode | 21 |
| 框架 | PyQt | 5 |

# 第6章 项目概览

## 6.1 系统总体功能结构

该基于PySide6的ping命令实现系统有以下几个功能

1. 提供命令行和图形界面两种操作方式
2. 提供发送ICMP数据包操作
3. 提供修改参数功能、如定义发送数据包数量、大小和超时时间

## 6.2 项目结构

该项目采用树形目录结构，采用MVC设计模式将模型、界面、控制器分别存放到不同目录。经典MVC模式中，M是指模型，V是视图，C则是控制器，使用MVC的目的是将M和V的实现代码分离，从而使同一个程序可以使用不同的表现形式。整体项目结构如图6-2所示。项目文件、目录及对应的作用如表6.1所示

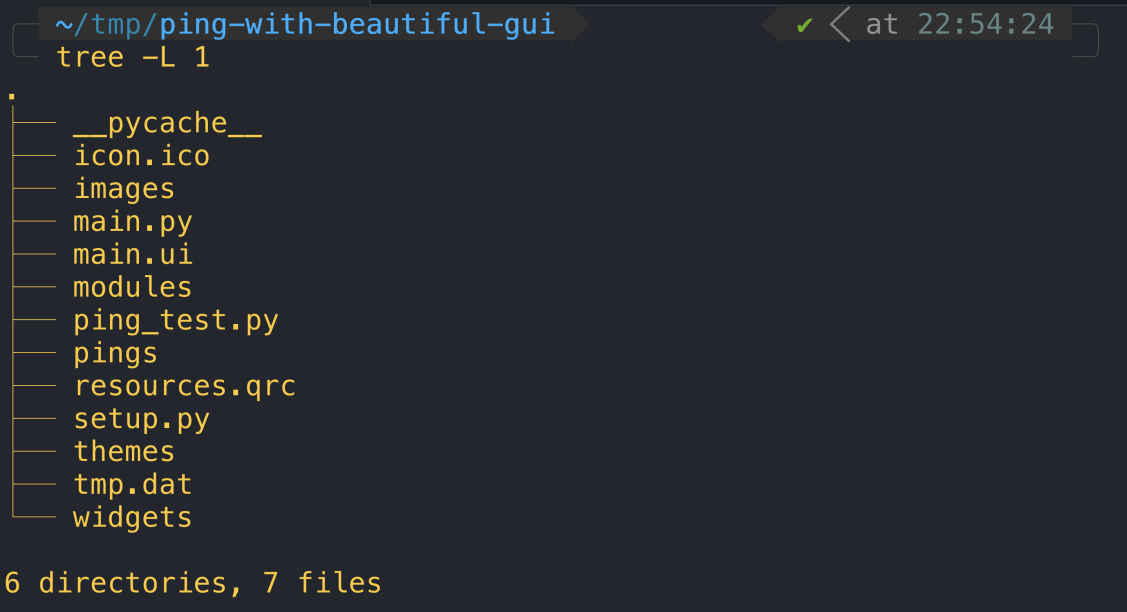


图 6-2 项目结构图

表 6.1 项目文件/目录分布表

|  |  |
| --- | --- |
| 目录/文件 | 作用 |
| pings | 套接字相关代码 |
| modules | 一些图形界面的组件和模块 |
| resource | 图形界面的资源文件 |
| theme | 图形界面的主题 |
| venv | python虚拟环境 |

# 第7章 系统设计

## 7.1 主要程序

### 7.1.1 Ping类

Ping类有几个方法来创建和发送ICMP数据包，接收和解析响应，并计算往返时间（RTT）的统计数据。

make\_socket方法为发送和接收ICMP数据包创建一个套接字。它将套接字选项设置为允许广播，并将TTL设置为1。它还将套接字绑定到一个随机端口，并将套接字的超时设置为超时属性中指定的值。

make\_packet方法创建了一个带有假头和随机字节有效载荷的ICMP数据包。它首先将校验和设置为0，并创建一个带有ICMP类型、代码、校验和、标识符和序列号的假头。然后，它生成一个随机字节的有效载荷，其长度在packet\_size属性中指定。它使用\_checksum方法计算报头和有效载荷的校验和，并用正确的校验和更新报头。最后，它以字节数组的形式返回包头和有效载荷。

\_checksum方法使用Internet Checksum算法计算一个字节数组的校验和。它将字节数组分割成16位字，并将它们相加。然后，它取和的一的补数，并以16位整数的形式返回。

发送方法使用make\_socket创建的套接字向目的地发送ICMP数据包。它首先调用make\_packet来创建数据包，并使用定时器函数记录发送时间。然后，它使用套接字的sendto方法将数据包发送到目的地，并返回发送时间。

接收方法从make\_socket创建的套接字中接收响应数据包。它首先使用选择方法等待数据包的到来，其超时时间等于超时属性。如果超时过后，它返回0和None。如果数据包到达，它使用\_parse\_icmp\_header方法解析ICMP头，并检查数据包标识符是否与make\_packet生成的标识符匹配。如果匹配，它使用\_parse\_ip\_header方法解析IP头，并返回接收时间、数据包大小、源IP地址、IP头和ICMP头。如果不匹配，它将更新超时并再次等待。

\_parse\_icmp\_header方法解析一个数据包的ICMP头，并将其作为一个字典返回，其键为 "type"、"code"、"checksum"、"packetid "和 "seqnumber"。

\_parse\_ip\_header方法解析了一个数据包的IP头，并将其作为一个字典返回，其键值为 "version"、"headerlen"、"tos"、"totallen"、"id"、"flags"、"fragoffset"、"ttl"、"protocol"、"checkum"、"srcip "和 "destip"。

ping方法向目的地发送多个ICMP数据包并接收它们的响应。它首先创建一个PingResponse对象来存储RTT统计数据和信息。然后，它使用发送方法向目的地发送数个数据包，使用接收方法接收它们的响应。它计算每个数据包的RTT并将其存储在一个列表中。它还计算最小、最大和平均RTT，并将它们存储在PingResponse对象中。最后，它将统计数据打印到控制台并返回PingResponse对象。

Ping命令UML，如图所示7-1。

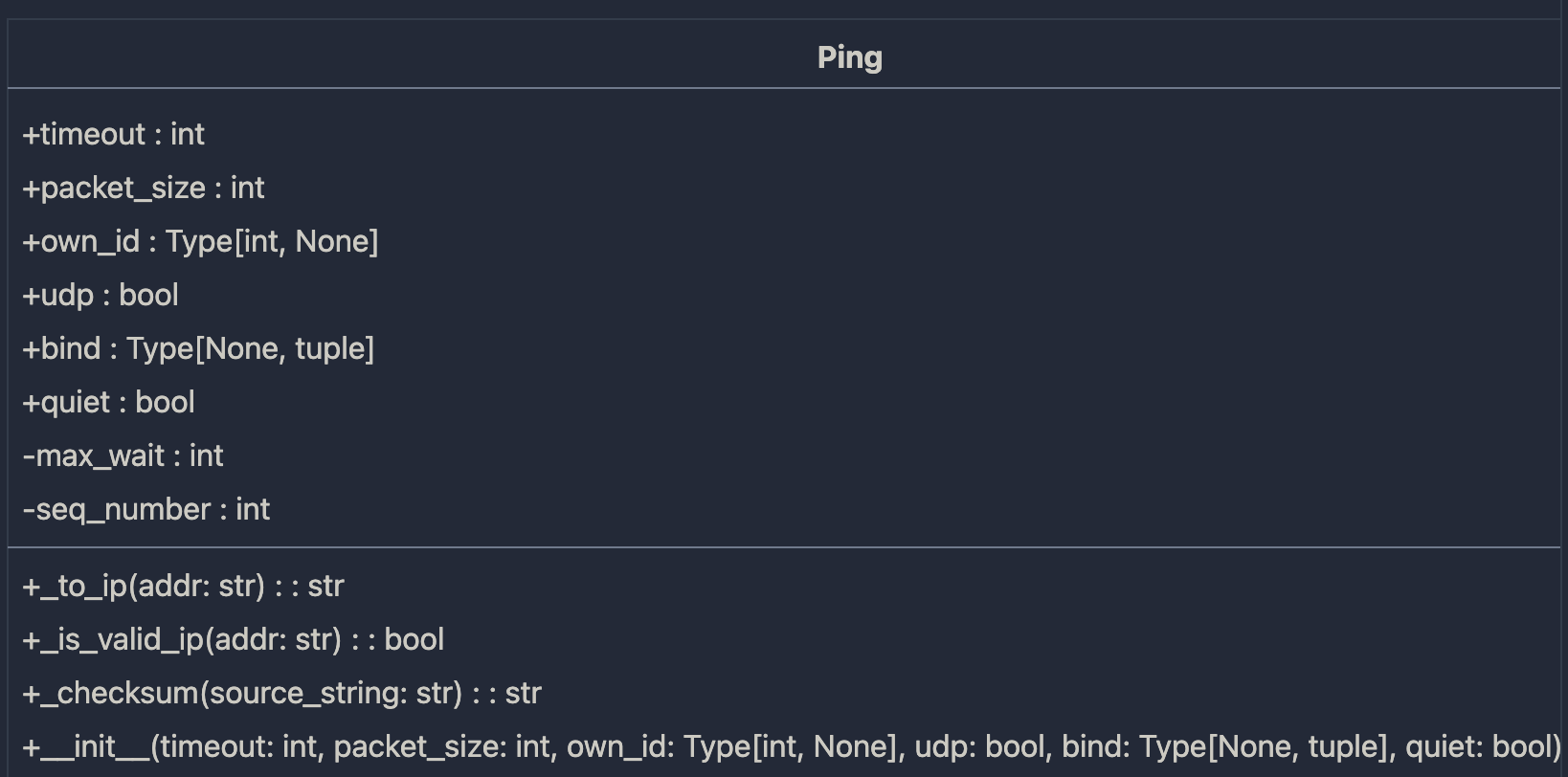


图 7-1 Ping UML图

关键代码如下：

from .consts import SUCCESS

class Response():

"""

Reponse of ping

"""

def \_\_init\_\_(self):

self.max\_rtt = None

self.min\_rtt = None

self.avg\_rtt = None

self.packet\_lost = None

self.ret\_code = None

self.messages = []

self.packet\_size = None

self.timeout = None

self.dest = None

self.dest\_ip = None

def \_\_str\_\_(self):

return str(self.to\_dict())

def to\_dict(self):

return {

"max\_rtt": self.max\_rtt,

"min\_rtt": self.min\_rtt,

"avg\_rtt": self.avg\_rtt,

"packet\_lost": self.packet\_lost,

"ret\_code": self.ret\_code,

"packet\_size": self.packet\_size,

"timeout": self.timeout,

"dest": self.dest,

"dest\_ip": self.dest\_ip,

}

def is\_reached(self):

return self.ret\_code == SUCCESS

def print\_messages(self):

for msg in self.messages:

print(msg)

### 7.1.2 PingResponse类

PingResponse类有用于存储RTT统计数据和消息的属性。rtt\_min、rtt\_max和rtt\_avg属性分别存储最小、最大和平均RTT。rtt\_list属性存储了RTT值的列表。packet\_loss属性存储了丢失数据包的百分比。messages属性存储了在ping操作中产生的控制台信息。

它是一个名为Response的大类的一部分，该类用于存储有关ping响应的信息。

Response类有几个属性，如max\_rtt, min\_rtt, avg\_rtt, packet\_lost, ret\_code, messages, packet\_size, timeout, dest, and dest\_ip。这些属性在构造函数中被初始化为None，以后可以设置。

to\_dict方法返回Response对象的一个字典表示。如果ret\_code属性等于SUCCESS，is\_reached方法返回True。print\_messages方法打印出存储在 messages 属性中的所有信息。

PingResponseUML图如7-2所示。

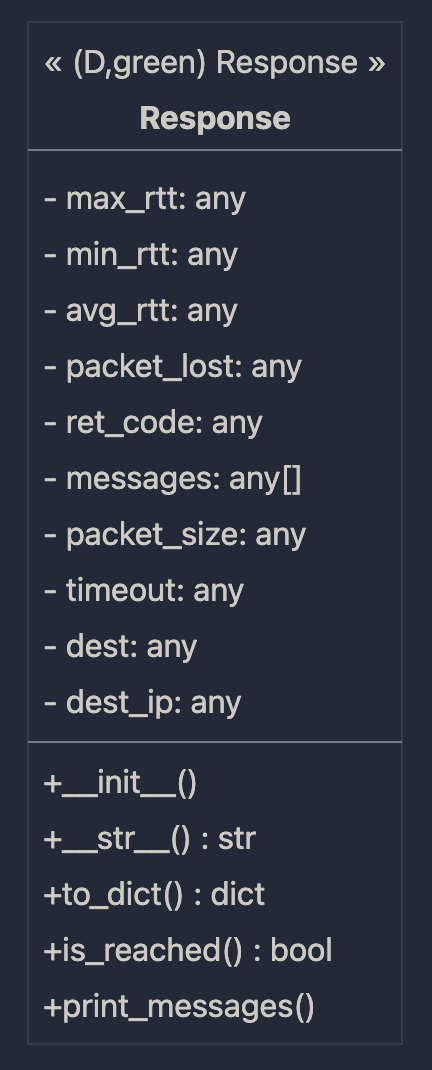


图 7-2 PingResponse UML图

关键代码如下:

class Ping():

def \_\_init\_\_(self, timeout=1000, packet\_size=55, own\_id=None, udp=False, bind=None, quiet=True):

self.timeout = timeout

self.packet\_size = packet\_size

self.own\_id = own\_id

self.udp = udp

self.bind = bind

self.quiet = quiet

if own\_id is None:

self.own\_id = os.getpid() & 0xFFFF

self.max\_wait = 1000 # ms

self.seq\_number = 0

# self.icmp\_echo\_reply = 0

self.icmp\_echo = 8

self.icmp\_max\_recv = 2048

def \_to\_ip(self, addr):

"""

If destination is not ip address, resolve it by using hostname

"""

if self.\_is\_valid\_ip(addr):

return addr

return socket.gethostbyname(addr)

def \_is\_valid\_ip(self, addr):

try:

socket.inet\_aton(addr)

except socket.error:

return False

return True

def \_checksum(self, source\_string):

"""

A port of the functionality of in\_cksum() from ping.c

Ideally this would act on the string as a series of 16-bit ints (host

packed), but this works.

Network data is big-endian, hosts are typically little-endian

"""

count\_to = (int(len(source\_string)/2))\*2

sum = 0

count = 0

# Handle bytes in pairs (decoding as short ints)

lo\_byte = 0

hi\_byte = 0

while count < count\_to:

if (sys.byteorder == "little"):

lo\_byte = source\_string[count]

hi\_byte = source\_string[count + 1]

else:

lo\_byte = source\_string[count + 1]

hi\_byte = source\_string[count]

try: # For Python3

sum = sum + (hi\_byte \* 256 + lo\_byte)

except: # For Python2

sum = sum + (ord(hi\_byte) \* 256 + ord(lo\_byte))

count += 2

## 7.2 图形界面设计

主程序主要为PyQt6/PySide6框架，通过调用和个模块接口进行功能的实现，主程序通过将按钮和\_clicked()系列函数进行绑定，从而实现不同按钮和对应事件的触发。

MainWindows UML如图7-3。

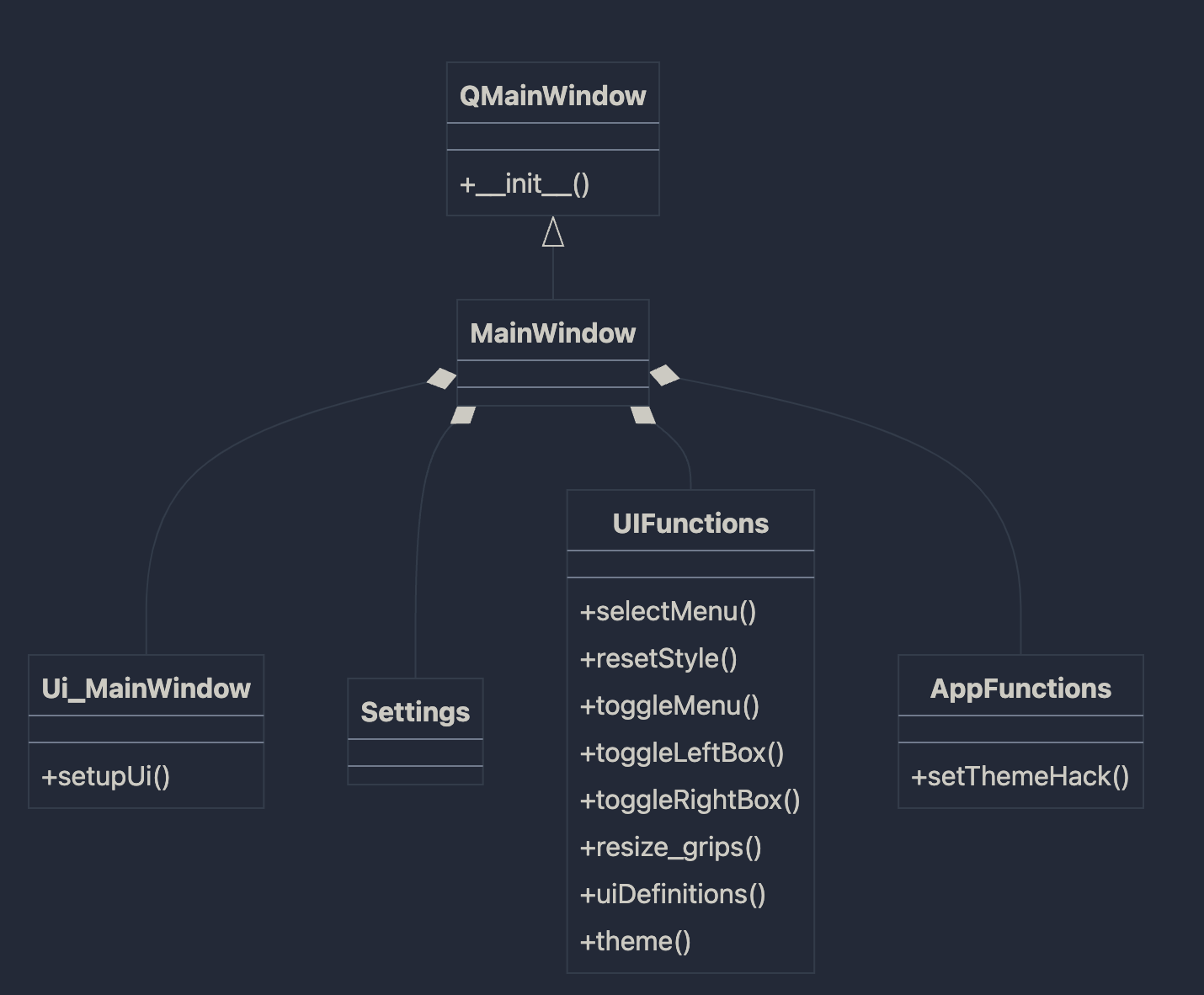


图7-3 MainWindow UML图

### 7.2.1 启动程序

该程序用于使用Qt Designer和PySide6构建的GUI应用程序。这个应用程序被称为 "PyDracula - Modern GUI"，它有一个基于Dracula for Python的颜色的主题。该脚本分别从 "modules "和 "widgets "目录中导入各种模块和部件。

定义了 MainWindow 类，它继承于 QMainWindow。\_\_init\_\_ 方法初始化了主窗口并设置了用户界面。Ui\_MainWindow类被用来设置用户界面，widgets变量被设置为用户界面部件的全局实例。

Settings.ENABLE\_CUSTOM\_TITLE\_BAR变量被设置为True，这样就可以使用自定义标题栏。应用程序的标题被设置为 "PyDracula - Modern GUI"，描述被设置为 "PyDracula APP - 基于Dracula for Python的颜色的主题"。

toggleButton部件被连接到一个切换菜单的lambda函数。uiDefinitions方法被调用来设置用户界面定义。

btn\_home、btn\_widgets、btn\_new和btn\_save部件被连接到buttonClick方法，它处理按钮的点击。btn\_start\_ping小组件连接到btn\_start\_ping\_clicked方法。

openCloseLeftBox和openCloseRightBox函数被定义为分别用来切换左边和右边的盒子。toggleLeftBox和settingsTopBtn部件被连接到这些函数。

show方法被调用以显示应用程序。

最后，useCustomTheme变量被设置为False，而themeFile变量被设置为 "themes\pydraculalight.qss"。如果useCustomTheme是True，主题方法被调用以加载和应用风格，setThemeHack方法被调用以设置黑客。stackedWidget被设置为主页，btn\_home widget被选中。

关键代码如下:

def btn\_start\_ping\_clicked(self):

# Ping(timeout=1000, packet\_size=55, own\_id=None, udp=False, bind=None, quiet=True)

timeout = int(self.ui.le\_timeout\_timeout.text().strip())

pkt\_size = int(self.ui.le\_pkt\_size.text().strip())

pkt\_cnt = int(self.ui.le\_pkt\_cnt.text().strip())

p = Ping(timeout=timeout, packet\_size=pkt\_size, own\_id=None, udp=False, bind=None, quiet=True)

url = self.ui.le\_url.text().strip()

self.ui.pte\_ping\_output.appendPlainText("starting pinging %s" % url)

# redirect output to variable

# buffer = StringIO

# sys.stdout = buffer

with open('tmp.dat', 'w') as sys.stdout:

res = p.ping(url, times=pkt\_cnt)

res.print\_messages()

with open('tmp.dat', 'r') as f:

# print\_output = buffer.getvalue()

data = f.read()

self.ui.pte\_ping\_output.appendPlainText(data)

# recovery output

# sys.stdout = sys.\_\_stdout\_\_

### 7.2.2 子启动程序

这是一个使用PySide6和Qt Designer创建的现代GUI应用程序。该应用程序使用了自定义标题栏和基于Dracula的颜色主题。在这个文件中，我们定义了一个MainWindow类，它继承自QMainWindow。在init方法中，我们设置了全局的widgets变量，它包含了我们在Qt Designer中创建的所有小部件。我们还设置了应用程序的标题和描述，并将其应用于窗口小部件。我们还定义了一些按钮单击事件，例如打开和关闭左侧和右侧的框，以及在单击按钮时切换页面。最后，我们设置了自定义主题和主题hack，并将主页设置为当前页面。app\_settings.py文件中的代码定义了一个Settings类，它将各种应用程序的设置作为类的属性来存储。这些属性指定了用于定制应用程序的GUI元素的值，如菜单宽度、左右框颜色、动画时间等等。

BTN\_LEFT\_BOX\_COLOR和BTN\_RIGHT\_BOX\_COLOR属性分别定义了左边和右边框的背景颜色。MENU\_SELECTED\_STYLESHEET属性存储了一个CSS样式表，用于对选定的菜单项进行造型。

你可以修改这些设置，根据你的喜好定制应用程序的GUI。它定义了一个叫做 "UIFunctions "的类，它继承了 "main.py "脚本中定义的 "MainWindow "类。

该脚本包含各种函数，如 "最大化还原"、"切换菜单"、"切换左框 "和 "切换右框"，它们用于管理和切换各种UI元素，如应用程序的窗口大小、左框和右框以及菜单项。startboxanimation "函数被盒子切换函数用来对盒子的大小变化进行动画处理。

此外，该脚本定义了选择或取消选择菜单项的函数，并使用 "主题 "函数应用外部文件中定义的自定义主题。还定义了几个全局变量，如 "GLOBALSTATE "和 "GLOBALTITLEBAR"，可以从代码库的其他部分访问或修改。

### 7.2.3 UI设计

使用Qtdesigner进行图形界面的设计，如图7-2所示。

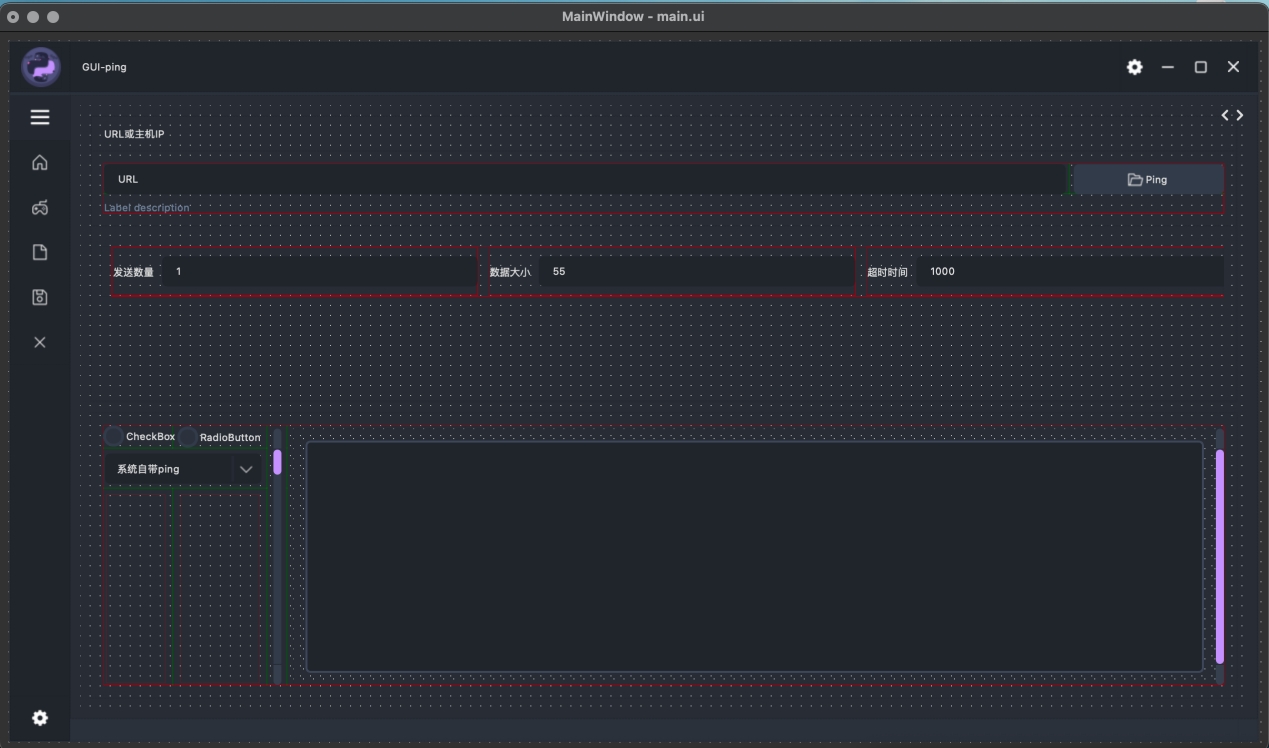


图 7-2 UI设计

Layout设计如图7-3，7-4所示。

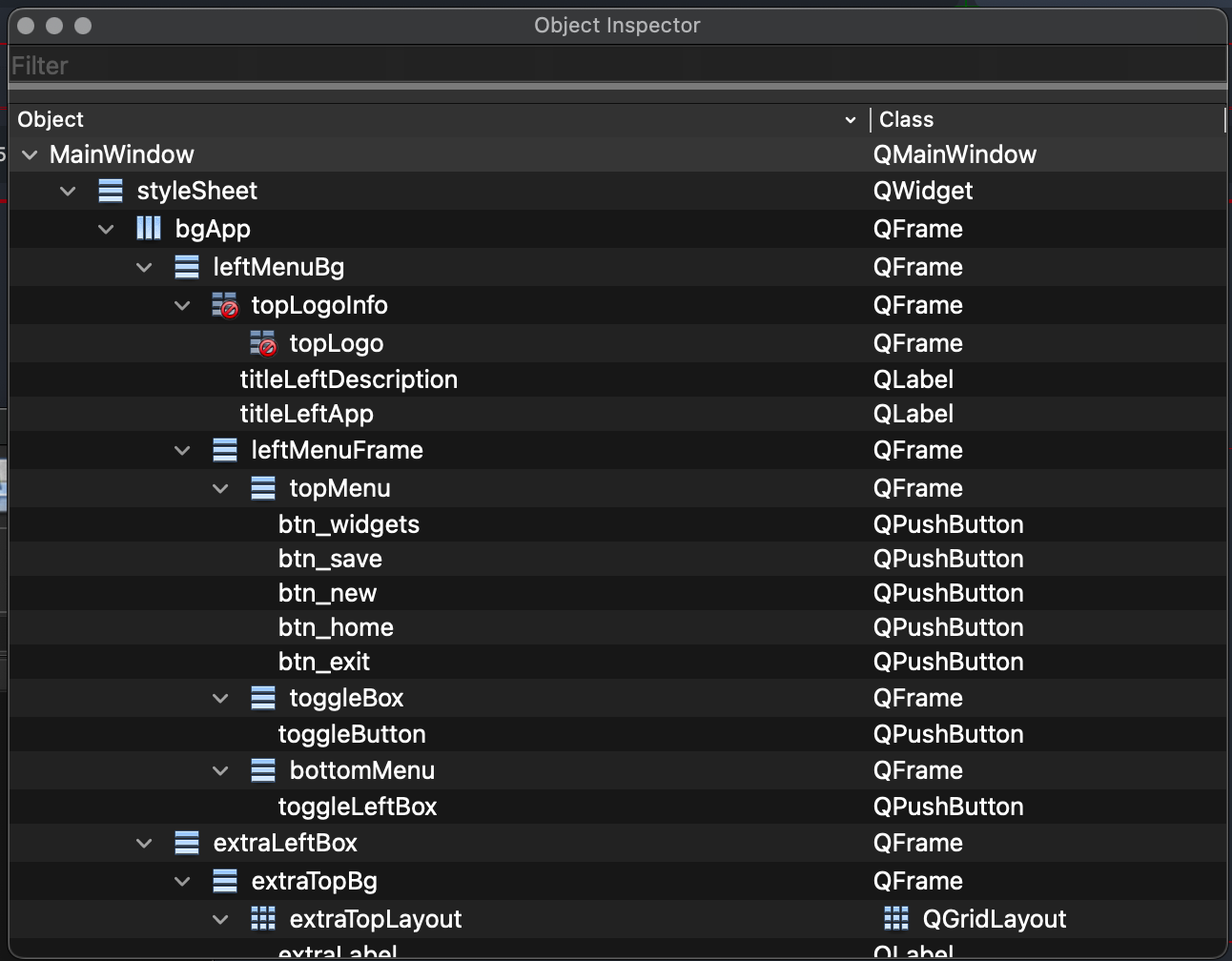


图 7-3 layout界面

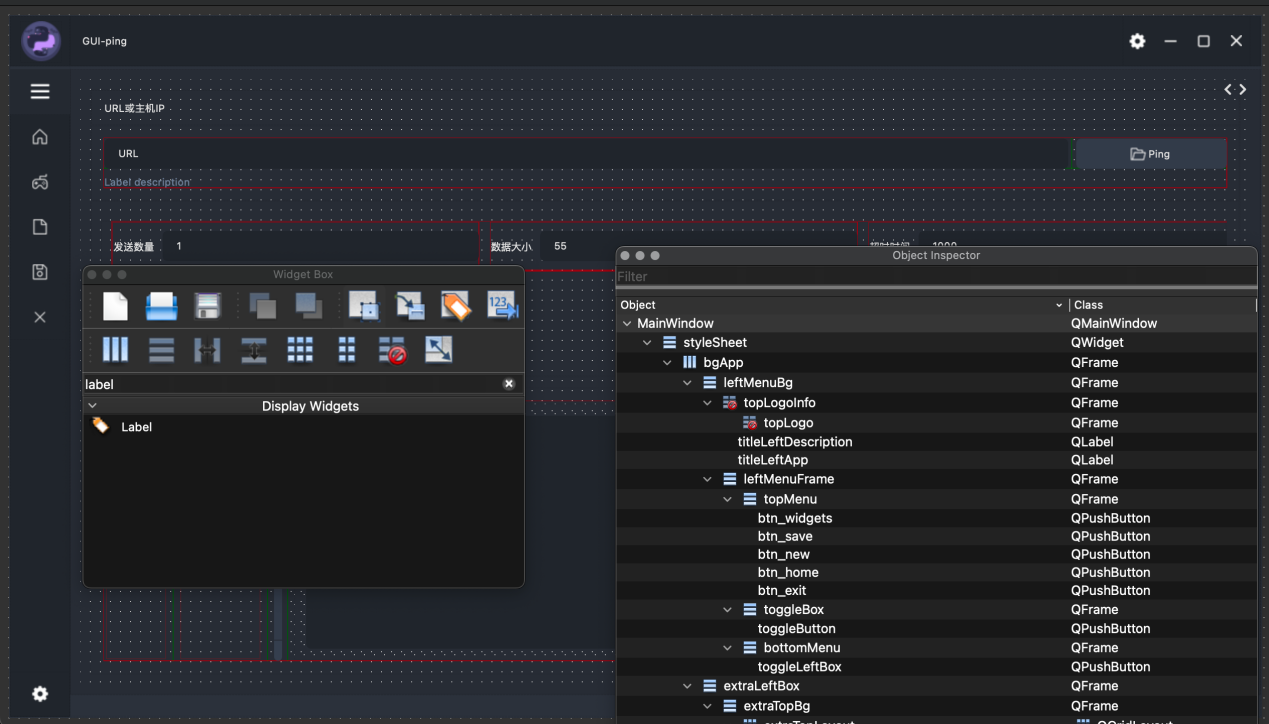


图 7-4

# 第8章 运行效果

## 8.1 主程序

主程序运行结果如图8-1，

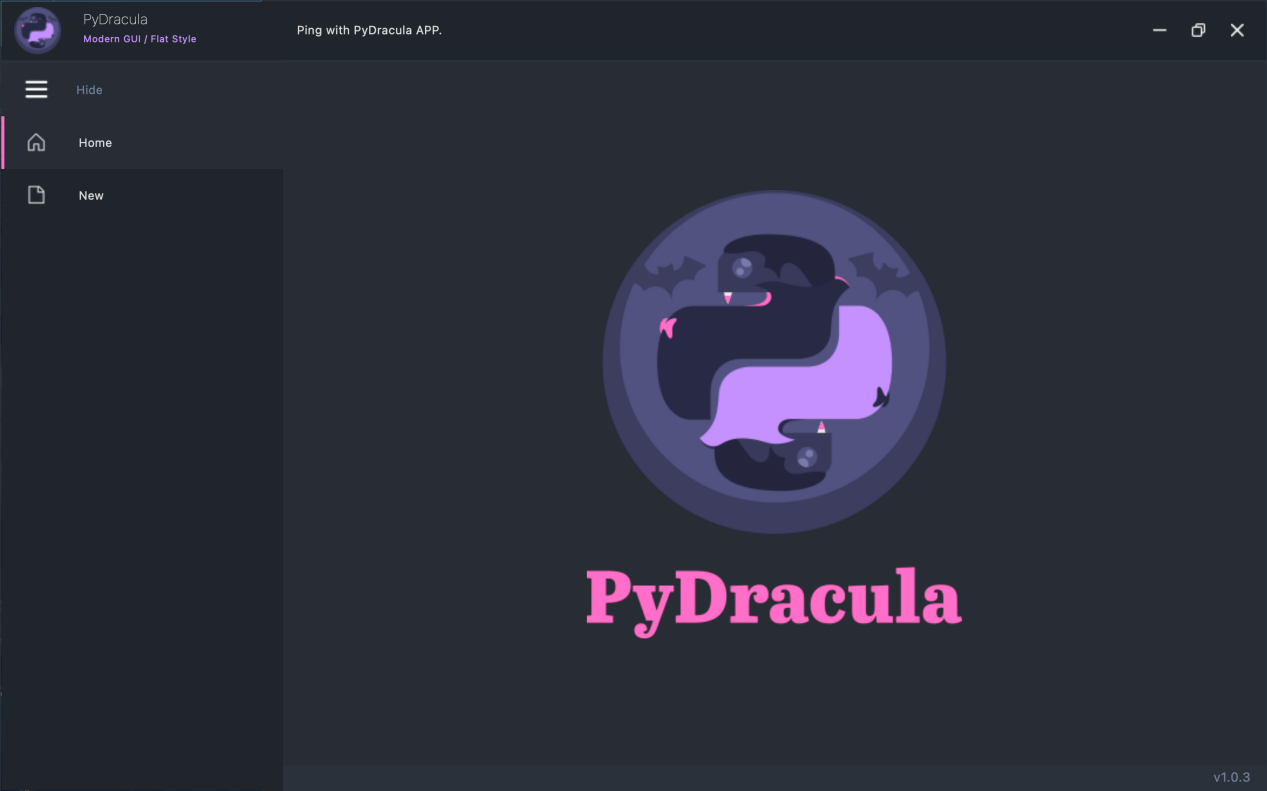


图 8-1 主界面

## 8.2 Ping程序页面

该界面可以让用户输入目的IP地址或者URL域名进行ping测试，同时提供了不同的参数如，ping数据包发送的数量、数据包大小、ping数据包超时时间，如图8-2所示。

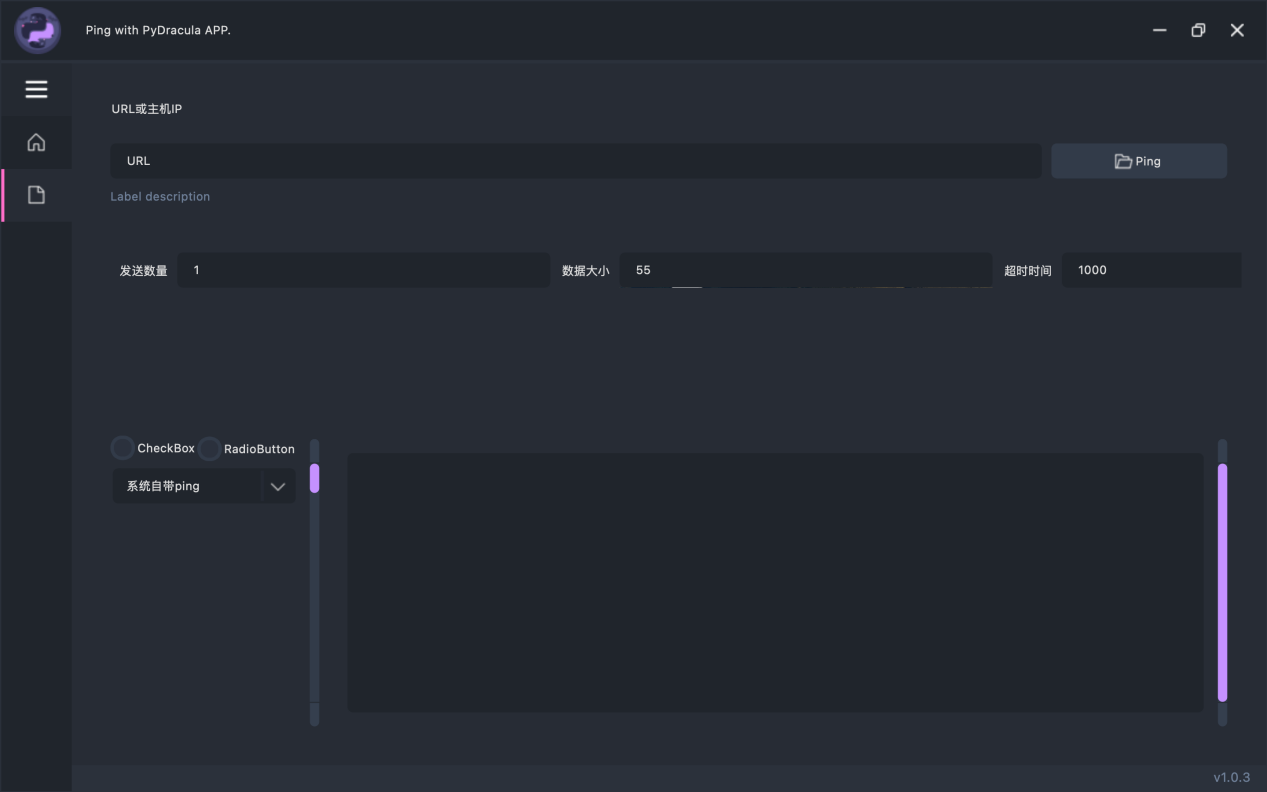


图 8-2 ping页面

## 8.3 Ping测试1

使用默认参数，发送一个数据包，payload大小为55字节，超时时间为1000毫秒。如图8-3所示。

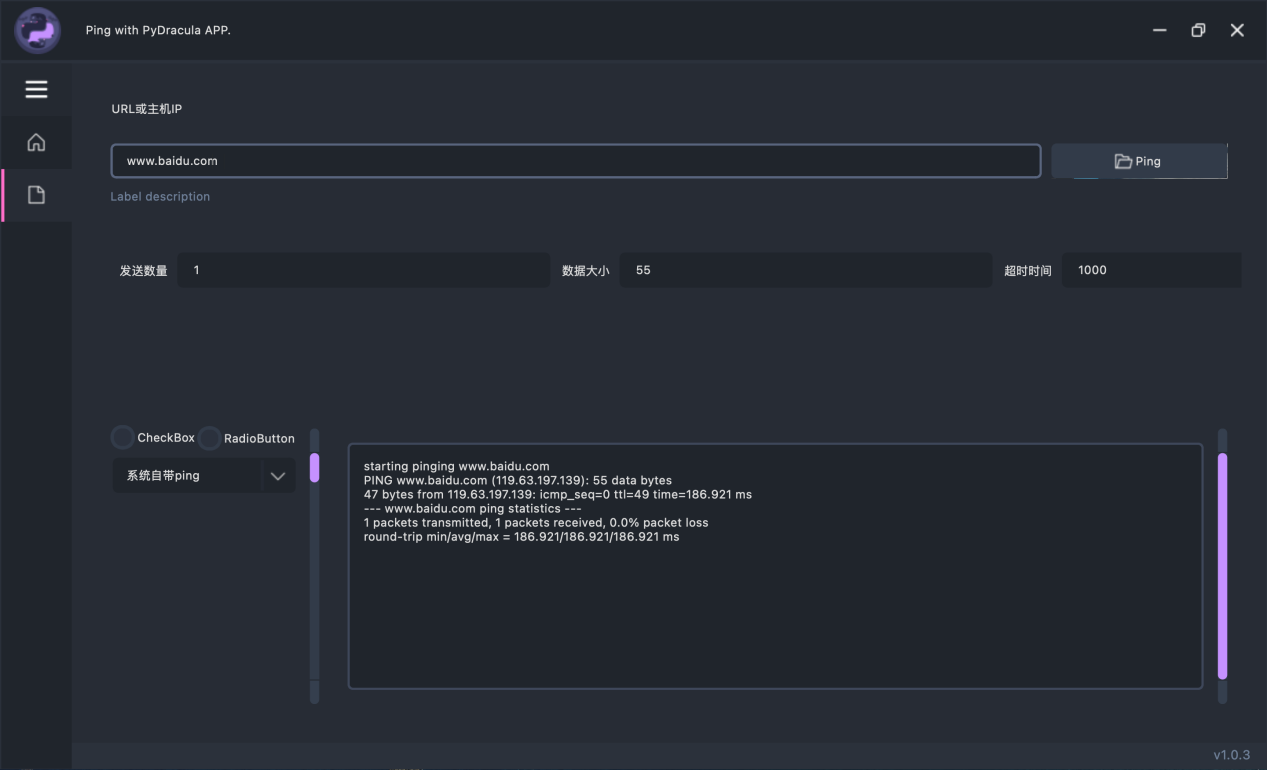


图 8-3 ping测试1

## 8.4 Ping测试2

目标URL地址为www.bing.com，发送ICMP数量为10，数据包大小为55，超时时间为1s，如图8-4所示。

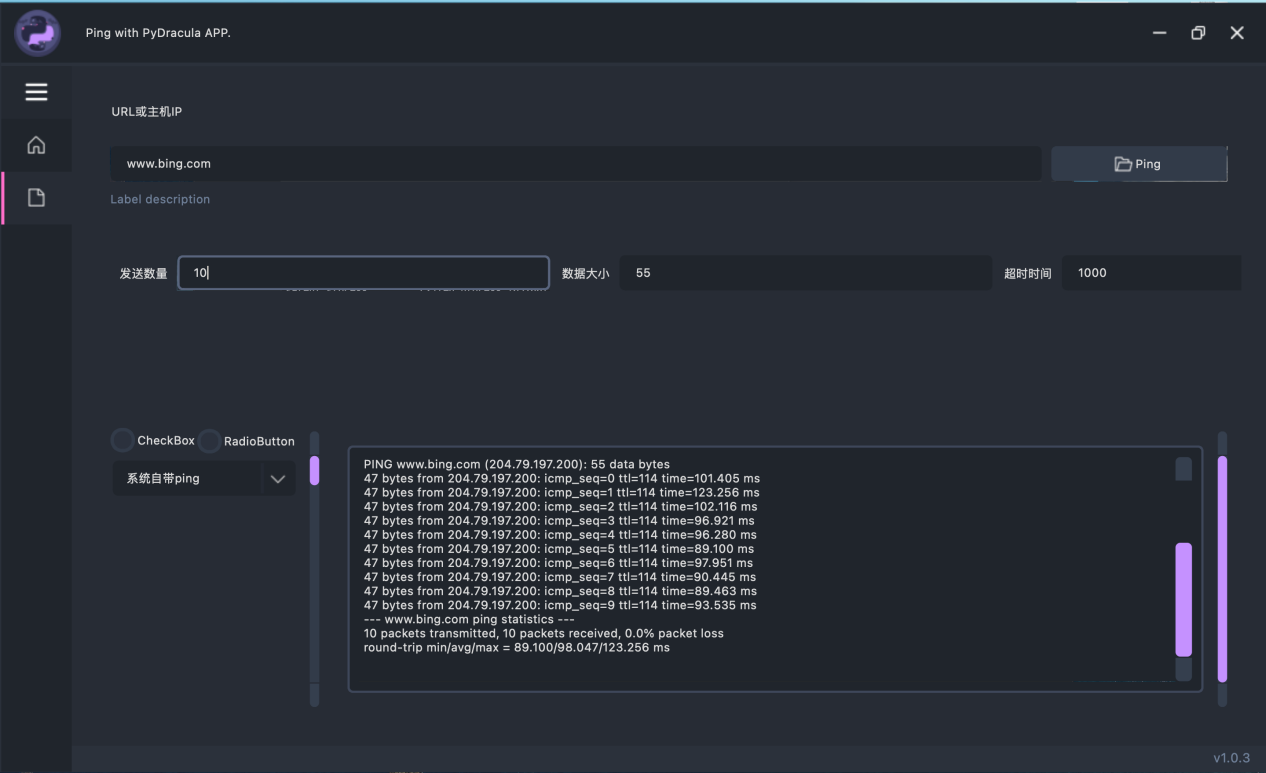


图 8-4 ping测试2

## 8.4 Ping测试3

目标IP地址为180.101.50.242，发送ICMP数量为5，数据包大小为100，超时时间为1s，如图8-5所示。

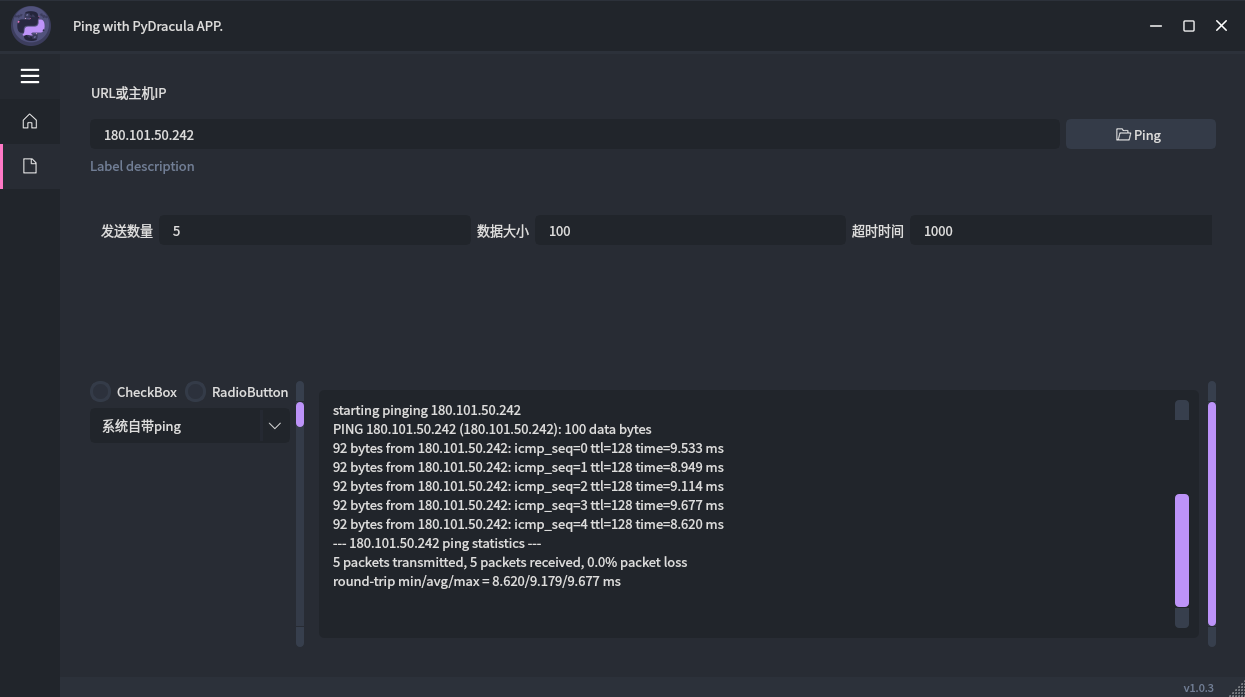


图 8-5 ping测试3

# 结论与展望

用PyQt实现Ping命令，使用原始套接字是可行的，为建立带有GUI的网络诊断工具提供了坚实的基础。该程序能够向目标主机发送ICMP数据包并接收ICMP Echo Reply数据包。该程序可以计算每个数据包的往返时间（RTT），并显示关于数据包丢失、RTT最小/最大/最小和TTL的详细统计数据。

未来的方向：

加强程序，以支持多个主机，并提供指定IP地址范围的能力来ping。

增加ping结果的图形表示，如显示RTT随时间变化的折线图或显示数据包丢失百分比的饼图。

研究用PyQt和原始套接字实现其他网络诊断命令的可能性，如Traceroute和Netcat，以提供一个完整的网络诊断套件。

优化程序以有效地处理大量的ICMP数据包，并减少网络拥堵或数据包丢失的影响。

探索使用该程序作为网络协议分析和反向工程的学习工具的潜力。

# 

# 参考文献

1. 谢希仁. 《计算机网络》. 第7版. 电子工业出版社. 2017年出版
2. Ping程序设计 : https://blog.csdn.net/zha\_ojunchen/article/details/106225905
3. python socket编程详细介绍 : https://blog.csdn.net/rebelqsp/article/details/22109925
4. python之struct详解 : https://blog.csdn.net/qq\_30638831/article/details/80421019
5. Ping 本质（ ICMP ） : https://www.jianshu.com/p/e1795962ad76
6. IP、ICMP、UDP、TCP 校验和算法 : <https://blog.csdn.net/weiweiliulu/article/details/20531367>
7. select Waiting for I/O completion : <https://docs.python.org/3/library/select.html#select.select>
8. IP Protocol Header Fundamentals Explained with Diagrams : <https://www.thegeekstuff.com/2012/03/ip-protocol-header/>
9. How to Calculate IP Header Checksum : [www.thegeeks/stuff.com/2012/05/ip-header-cheade](http://www.thegeeks/stuff.com/2012/05/ip-header-cheade)
10. 明日科技《PyQt从入门到精通》，清华大学出版社. 2019年出版 120-129.

# 

# 致谢

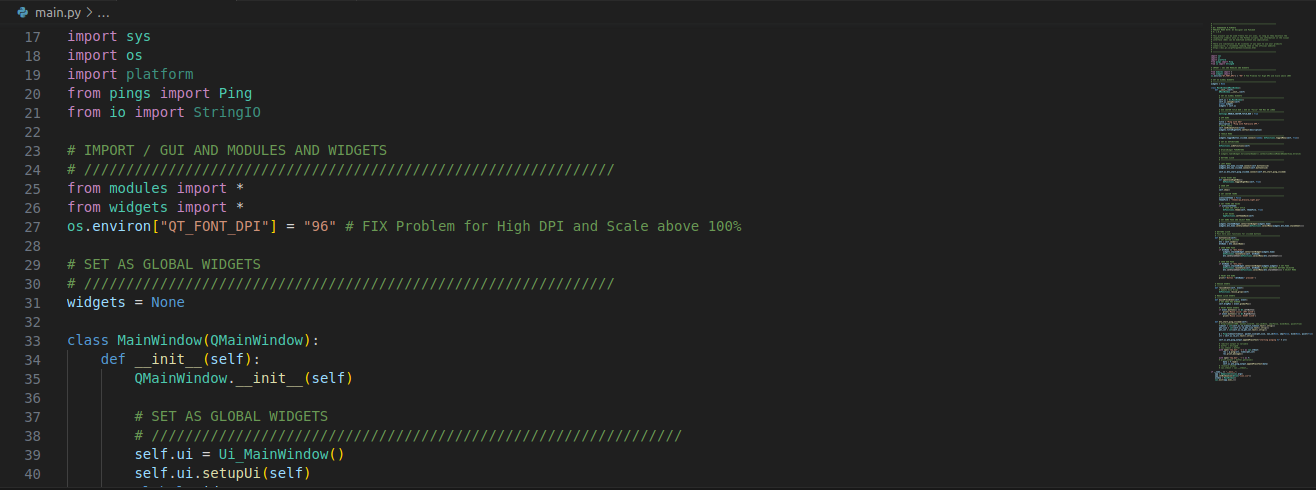
大学四年的生活即将结束，这四年过得真的好快，仿佛自己昨天还是那个懵懵懂懂的大一新生，如今我也准备从我的母校毕业，我的人生也即将开始新的篇章。我非常感谢母校这些年来对我的悉心栽培，在母校所学习到的不仅仅是知识，还有为人处世的道理，我同样也很感谢我的同学和老师对我的支持与帮助。致谢部分我最想感谢的还是我的父母，在我的学习成长道路上若是没有父母的陪伴与支持我不会有今天这样的成就，养育之恩，无法回报，希望你们一直健健康康开开心心。

写这篇论文一开始就遇到了很多困难，写的过程中也是曲折不断，这期间也是经历了很长的时间以及数次改动，最终我的这篇论文是在我的导师刘卫江老师的不断的耐心指导下完成的，我也是非常的感谢刘老师可以那么有耐心的帮助我，每次我有不懂的问题刘老师都会帮我解答，我也是由衷的佩服刘老师知识储备如此丰富，感觉跟百科全书一般。在刘老师的栽培下我学习到了很多东西，比如说严谨的精神，这对于我以后的人生道路而言是一笔非常宝贵的财富。

最后，我还是想感谢每一位曾经教导和帮助过我的老师和同学，没有老师和同学的帮助我在学校的学习也不会那么顺利，也不会掌握那么多知识、我会继续努力加油，用我的努力去实现我的人生价值。

**附录**

部分主要代码片段及构成如附图1、附图2、附图3、附图4所示



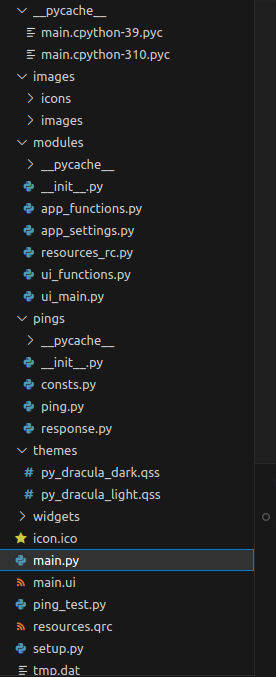
附图1 main.py 部分代码



附图2 ping.py 部分代码



附图3 ui\_main.py 部分代码



附图4 工程组成部分截图